VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

SOFISTIKOVANÉ STOPKY PRO MĚŘENÍ ČASŮ NA ZÁVODECH

SOPHISTICATED STOPWATCH FOR MEASURING TIMES IN COMPETITIONS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **ONDŘEJ SKRYJA**
AUTHOR

VEDOUCÍ PRÁCE **prof. Ing. ADAM HEROUT, PhD.**
SUPERVISOR

BRNO 2018

Zadání bakalářské práce/21015/2017/xskryj02

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav počítačové grafiky a multimédií Akademický rok 2017/2018

Zadání bakalářské práce

Řešitel: **Skryja Ondřej**
Obor: Informační technologie
Téma: **Sofistikované stopky pro měření časů na závodech**
Sophisticated Stopwatch for Measuring Times in Competitions
Kategorie: Uživatelská rozhraní

Pokyny:

1. Seznamte se s problematikou návrhu a vývoje mobilních aplikací; zaměřte se na platformu Android.
2. Vyhledejte a analyzujte existující aplikace pro měření času ve sportu.
3. Navrhněte a prototypujte způsob interakce s aplikací a jednotlivé prvky uživatelského rozhraní.
4. Navrhněte a implementujte řešenou aplikaci pro pokročilé měření času na sportovních závodech.
5. Testujte vytvořenou aplikaci na uživatelích a iterativně ji vylepšujte.
6. Zhodnoťte dosažené výsledky a navrhněte možnosti pokračování; vytvořte plakátek a krátké video pro prezentování projektu.

Literatura:

- Steve Krug: Don't Make Me Think, Revisited: A Common Sense Approach to Web Usability, ISBN-13: 978-0321965516
- Android Developers: https://developer.android.com/index.html
- Susan M. Weinschenk: 100 věcí, které by měl každý designér vědět o lidech, Computer Press, Brno 2012

Pro udělení zápočtu za první semestr je požadováno:

- Body 1 až 3, značné rozpracování bodů 4 a 5.

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Herout Adam, prof. Ing., Ph.D.**, UPGM FIT VUT
Datum zadání: 1. listopadu 2017
Datum odevzdání: 16. května 2018

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav počítačové grafiky a multimédií
L.S. 612 66 Brno, Božetěchova 2

doc. Dr. Ing. Jan Černocký
vedoucí ústavu

Abstrakt

V této bakalářské práci se budu zabývat návrhem mobilní aplikace pro organizaci závodů. Cílem aplikace je zjednodušit organizaci závodníků, jejich rozdělení do jednotlivých kategorií a změření času. Aplikace je vytvářena pro operační systém Android. Seznámím čtenáře se základními komponenty Android aplikací a vývojovým prostředím Android Studio. Detailně se budu věnovat návrhu aplikace a jednotlivým funkcím.

Abstract

In this bachelor thesis I will deal with the design of a mobile application for organizing competitions. The main goal of the application is to simplify the organization of contestants, their division into individual categories and time measurement. The app is created for the Android operating system. I will introduce the basic Android components and Android Studio development environment. In detail, I will devote myself to application design and individual functions.

Klíčová slova

Android, Kotlin, Závod, Stopky, Měření času, Mobilní aplikace, Google Drive, Google Sheets

Keywords

Android, Kotlin, Competition, Stopwatch, Time measurement, Mobile application, Google Drive, Google Sheets

Citace

SKRYJA, Ondřej. *Sofistikované stopky pro měření časů na závodech.* Brno, 2018. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce prof. Ing. Adam Herout, PhD.

Sofistikované stopky pro měření časů na závodech

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana profesora Adama Herouta. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Ondřej Skryja
10. května 2018

Poděkování

Rád bych poděkoval vedoucímu mé bakalářské práce, prof. Ing. Adamovi Heroutovi, Ph.D, za jeho odborné rady a vedení. Dále děkuji rodině za okamžitou kritickou zpětnou vazbu při vývoji aplikace a všem, kteří se podíleli na testování nově vytvořené aplikace.

Obsah

1 Úvod **3**

2 Aplikace pro měření času **4**
2.1 Sport Time Keeper 4

3 Prvky Android aplikace **6**
3.1 Android studio 6
3.2 Struktura projektu 6
3.3 Activity 7
3.4 View a ViewGroup 8
3.5 Layout 9
3.6 TextView, EditText a Button 10
3.7 Adapter 11
3.8 RecyclerView 11
3.9 Spinner 12
3.10 Toolbar 13
3.11 Options menu 13
3.12 Navigation Drawer 14
3.13 Toast 15

4 Uživatelé **16**
4.1 Charakteristika uživatele 16
4.2 Potřeby uživatele 16

5 Návrh aplikace **18**
5.1 O testování 18
5.2 Propojení entit 19
5.3 Základní rozložení 20
5.4 Karta času 21
5.5 Výpočet času a vyhodnocení závodu 23
5.6 Závody 24
5.7 Závodníci 25
5.8 Kategorie 27
5.9 Přiřazení závodníka k času 28
5.10 Startovní a výsledková listina 30
5.11 Google Drive a Google Sheets 30

6 Implementace **32**

6.1 Kotlin vs. Java 32
6.2 Ukládání dat 33
6.3 Autentizace služeb Google 33

7 Závěr **34**

Literatura **35**

Kapitola 1

Úvod

Cílem práce je vytvoření mobilní aplikace pro operační systém Android. Aplikace slouží ke zjednodušení organizace a měření času na menších závodech nebo trénincích. Hlavní funkcí aplikace je měření času závodníkům. Závodníky je předem nutno rozdělit do kategorií a nastavit čas startu. Závodníci mohou startovat jak hromadně, tak i intervalově, tedy s předem danými časovými rozestupy. Při průchodu závodníka cílem bude muset uživatel pomocí stisku tlačítka na obrazovce telefonu vytvořit nový čas. K tomuto času následně přiřadí závodníka. Výsledný čas závodníka je tedy určen časem startu závodníka a časem startu závodu. Výsledky je možné uložit do Google Sheets[1] a tak snadno sdílet s účastníky závodu.

Hlavním důvodem pro vytvoření mobilní aplikace je možnost uspořádat závod nebo trénink bez nutnosti připojení k elektrickému proudu nebo síti internet. S touto základní myšlenkou je celá aplikace tvořena. Nesmí být závislá na mobilních sítích, musí být spolehlivá a jednoduše ovladatelná i v nepříznivých podmínkách (například za deště nebo naopak přímého slunečního svitu).

Pro vytvoření nativní Android aplikace je možno vybrat si ze dvou programovacích jazyků. Java, první nativně podporovaný jazyk pro vývoj Android aplikací, nebo Kotlin, který byl přidán mezi podporované jazyky v roce 2017[8]. Naše aplikace je psána v jazyce Kotlin. Důvodů je několik a více se jim budu věnovat v jedné z následujících kapitol.

Součástí zadání této práce je i vytvoření prezentačních materiálů, konkrétně plakátek a krátké video. Jejich úkolem je stručné a výstižné představení vytvářené aplikace.

[1] https://www.google.cz/intl/cs/sheets/about/

Kapitola 2

Aplikace pro měření času

V současné době je v digitálním obchodě Google Play[1] (primární obchod s aplikacemi systému Android) velké množství aplikací pro měření času. Tyto aplikace se dají rozdělit do několika skupin podle nabízených funkcí.

1. Základní stopky

Měří čas, případně kola, pouze pro jednu osobu. Nezvládají tedy měřit více časů zároveň. Většinou se jedná o velmi jednoduché aplikace. Můžeme je najít takřka v každém Android zařízení.

2. „Běžecké“ stopky

Stopky jsou primárně určené pro dráhové sporty. Zvládají změřit 1-4 časy (lze ovšem najít aplikace, které zvládají změřit například až 8 časů). Zpravidla lze měřit i časy kola. Stále ale limitují uživatele maximálním počtem závodníků.

3. Stopky s „neomezeným“ počtem závodníků

Kategorie s nejmenším počtem zástupců (při prohledávání Obchodu Play jsem jich našel pouze několik). Jedná se o stopky, které zvládají měřit i více než 8 časů zároveň. Většinou je maximální počet omezen jak designem aplikace, tak i monetizačním modelem (například odemčení více slotů pro závodníky pomocí platby).

Naše aplikace se řadí právě do třetí kategorie. Řeší problém měření času velkého (omezeno pamětí zařízení) množství závodníků a jejich dělení do kategorií.

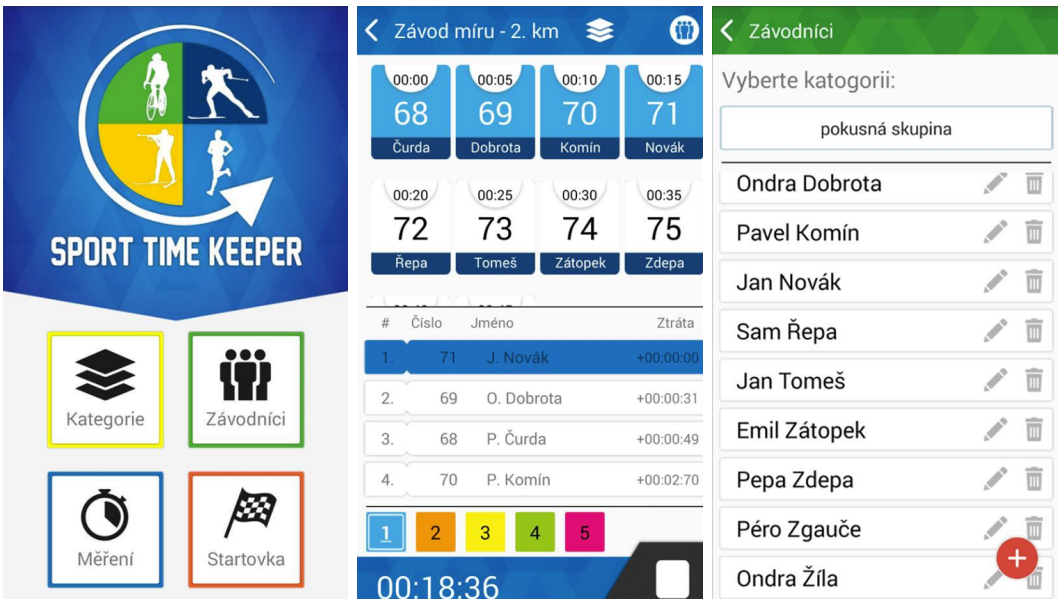
2.1 Sport Time Keeper

Sport time keeper[2] je jedna z aplikací, které řeší stejný problém. Aplikace je v Google Play zdarma, ale je omezena na dvě kategorie po sedmi závodnících. Za poplatek je možné odemknout neomezený počet kategorií anebo závodníků. Základní funkce jsou velmi podobné. Jak naše stopky, tak i Sport time keeper ukládá závody, seznamy závodníků a kategorií. Navíc umožňuje měřit časy na kola. Neumožňuje však měřit čas více kategorií najednou.

[1] https://play.google.com/store

[2] https://play.google.com/store/apps/details?id=cz.appyours.sporttimekeeper

Sport Time Keeper zobrazuje pro zaznamenání času jedno tlačítko pro každého závodníka (v odstartované kategorii), které slouží k zaznamenání času (obrázek 2.1 uprostřed). Tato varianta zaznamenávání času je velmi intuitivní a rychlá, ale pouze v případě malého počtu závodníků. Zároveň je nutné znát jméno nebo startovní číslo závodníka už v okamžiku pořízení času. Další problém může nastat u úpravy nebo smazání závodníka v seznamu závodníků (obrázek 2.1 vpravo). Zde je tlačítko pro přidání závodníka, které bude při větším počtu účastníků částečně překrývat důležité ovládací prvky. Dále jsem nenašel způsob jak opravit čas (při brzkém nebo pozdním zaznamenání) anebo změnit závodníka (při záměně jména nebo startovního čísla). Tyto nedostatky se budu snažit vyřešit odlišným způsobem pořízení času, kterému se budu věnovat v jedné z dalších kapitol.



Obrázek 2.1: Na prvním obrázku (vlevo) je úvodní stylizovaná obrazovka aplikace Sport Time Keeper. Druhý obrázek (uprostřed) zobrazuje odstartovaný závod. V horní části jsou tlačítka závodníků (takovýto způsob zadávání časů nebude pro naši aplikaci vhodný), ve spodní pak pořadí v jednotlivých kolech a aktuální čas závodu. Na třetím obrázku (vpravo) je seznam závodníků. Dle mého názoru je zde nevhodně zvolené tlačítko pro přidání závodníka (červené tlačítko).

Kapitola 3

Prvky Android aplikace

V této kapitole se seznámíme se základními prvky, které tvoří Android aplikaci. Popíši zde hlavně ty komponenty, které jsou následně použity při tvorbě našich stopek. Mějme ale na paměti, že mnohé komponenty jsou dosti složité a cílem této práce není jejich hloubková analýza, proto bude jejich popis zjednodušený a nebudou popsány všechny jejich funkce. Podrobnější informace je možné nalézt v online dokumentaci[1].

Obrázky v této kapitole slouží hlavně pro lepší představu o popisovaných prvcích. Obrázky 3.2, 3.4, 3.5, 3.6 a 3.7 jsou převzaty z dokumentace[1].

3.1 Android studio

Android Studio[1] je oficiálně podporované IDE[2] pro vývoj nativních Android aplikací. Umožňuje vývoj pro všechny typy android zařízení od telefonů a tabletů po Android TV a Android Auto. Sdružuje prostředí pro psaní kódu v Javě nebo Kotlinu, prostředí pro návrh aplikace a také obsahuje emulátor Android zařízení. Android Studiu se podrobně věnuje například kniha Expert Android Studio[11]

Jako mnohá moderní vývojová prostředí, tak i Android Studio pomáhá svému uživateli například našeptáváním a dokončováním slov, generováním částí kódu nebo analýzou kódu v reálném čase a okamžitým upozorněním na chyby. Dovoluje instalaci, debug či profilování jak v emulátoru, tak i v připojeném fyzickém zařízení (to musí být ve vývojářském módu). Pokud máme připojeno zařízení s Android 5.0 (API 21) nebo vyšším, můžeme použít takzvaný „Instant run“[3]. To dovoluje aplikovat změny (pokud nejsou příliš rozsáhlé) bez nutnosti přeložení a následné instalace celé aplikace.

Samotný emulátor pak zvládá instalovat a spouštět aplikace jako na reálném telefonu. Dovoluje vyzkoušet velké množství hardwarových i softwarových konfigurací, nastavovat sítě a senzory nebo simulovat příchozí hovory a SMS.

3.2 Struktura projektu

Projekt v Android Studiu lze zobrazit několika způsoby. Projekt si můžeme zobrazit jako „Project“, který zobrazuje veškeré soubory a složky v adresáři projektu. Dále můžeme použít zobrazení „Package“, které nám projekt zobrazí po jednotlivých použitých balících. Je

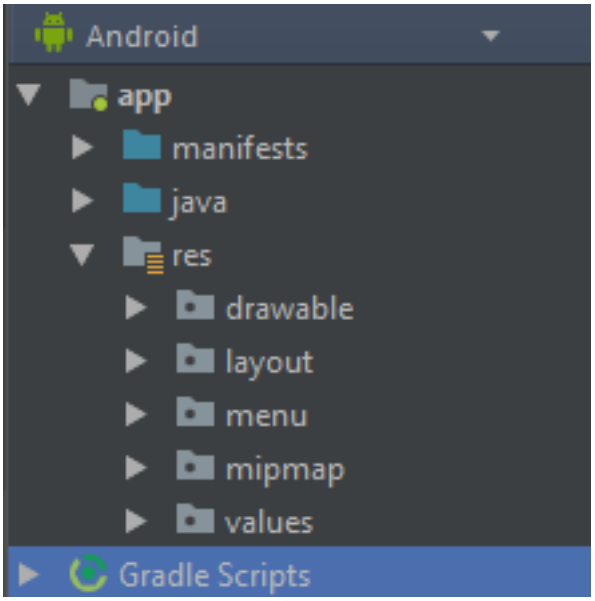
[1] https://developer.android.com/studio/index.html
[2] Integrated Development Environment – Integrované Vývojové Prostředí
[3] https://developer.android.com/studio/run/index.html#instant-run

možné využít několik dalších zobrazení. Z mého pohledu je ovšem nejpoužívanější zobrazení „Android“ (obrázek 3.1).

V kořenové úrovni se nachází položka „Gradle Scripts“ obsahující skripty pro nástroj Gradle[4]. Gradle zde slouží zejména pro automatické stahování knihoven a jejich závislostí a pro nastavení verze vydávané aplikace. Je ale možné využít i všechny jeho další možnosti.

Druhou položkou je „app“. V ní najdeme „manifest“[5] definující potřebná povolení, jednotlivé komponenty aplikace, jméno balíku a potřebné softwarové a hardwarové funkce. Dále zde najdeme položku „java“. Zde jsou všechny zdrojové a testovací soubory. Mohou být napsané jak v Javě (první nativně podporovaný jazyk – proto pojmenování „java“), nebo Kotlinu.

Velmi důležitou částí je položka „res“[6] (resources) obsahující veškeré soubory popisující vzhled aplikace (pokud není definovaný ve zdrojovém kódu, což se typicky nepoužívá). Jsou zde XML soubory popisující rozložení a styl jednotlivých komponent na obrazovce, položek menu nebo hodnoty řetězců pro lokalizaci a podobně. V neposlední řadě jsou zde také drawable[7] – vektorové (XML, SVG, PSD) nebo rastrové obrázky (PNG, JPG, GIF).



Obrázek 3.1: Struktura Android projektu se zobrazením „Android“

3.3 Activity

Aktivita[8] (Activity) je základním stavebním kamenem každé aplikace. Zprostředkovává interakci s uživatelem a jejím prostřednictvím jsou zobrazeny informace a ovládací prvky na obrazovce. Aktivita zpravidla rozšiřuje (override) metodu „onCreate“, která specifikuje, co se stane při vytvoření aktivity. Využívá se ke vložení view, inicializaci proměnných anebo nastavení komponent ve view dle aktuální situace. Často se také využívá metoda „onPause“,

[4] https://gradle.org/
[5] https://developer.android.com/guide/topics/manifest/manifest-intro.html
[6] https://developer.android.com/guide/topics/resources/index.html
[7] https://developer.android.com/guide/topics/resources/drawable-resource.html
[8] https://developer.android.com/reference/android/app/Activity.html

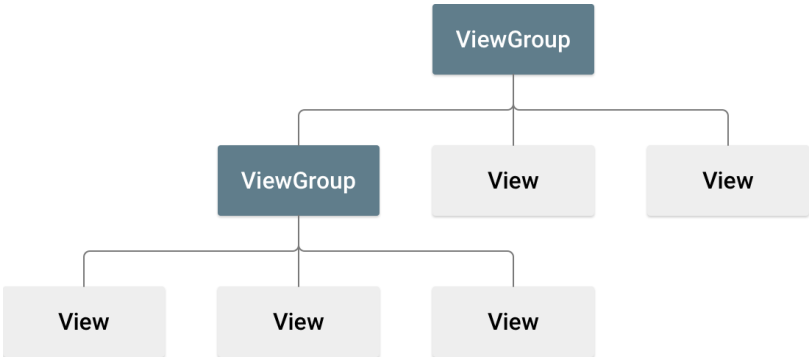
která se vykoná, pokud se má do popředí dostat jiná aktivita (například při vytvoření nové aktivity nebo otevření jiné aplikace). Dále metoda „onResume“, která se vykoná při návratu do metody. V rámci životního cyklu aktivity je možné využít i další metody. Každá nově vytvořená aktivita je ukládána do zásobníku aktivit. Aktivita na vrcholu zásobníku je aktivní aktivita.

Pro vytvoření nové aktivity je třeba připravit Intent[9] a pomocí metody „startActivity“ ji spustit. Intent je abstraktní popis operace, která má být provedena. Slouží zejména pro předání informace o tom, jaká aktivita se má spouštět, a dále k předání dat do této aktivity.

3.4 View a ViewGroup

Aktivita většinou obsahuje alespoň jeden nebo i více objektů třídy View[10]. View jsou umístěna ve stromové struktuře (obrázek 3.2). Kořenové view bývá zpravidla ViewGroup[11] (podtříta třídy View), která je základní třídou pro Layout, o kterém budu psát v následující části.

Na view můžeme pohlížet jako na viditelné prvky aplikace, které dovolují uživateli s aplikací interagovat. ViewGroup jsou pak neviditelné kontejnery, které určují vzájemnou polohu jednotlivých view.



Obrázek 3.2: Znázornění hierarchie view definující layout.

Každému view v aktivní aktivitě můžeme nastavit jeho parametry, „focus“ (view očekává uživatelský vstup), viditelnost (visibility) nebo listenery (listeners). Listenery vykonají definovanou činnost v případě, že jsou spuštěny akcí, které naslouchají. Díky tomu můžeme například nastavit chování tlačítka při stisku. Nejčastěji se struktura view definuje jako XML soubor obsahující základní layout (kořenové ViewGroup), uvnitř něhož jsou další zanořené view. Pro odkazování se na konkrétní view použijeme parametr „id“, jehož hodnota musí být unikátní.

[9] https://developer.android.com/reference/android/content/Intent.html
[10] https://developer.android.com/reference/android/view/View.html
[11] https://developer.android.com/reference/android/view/ViewGroup.html

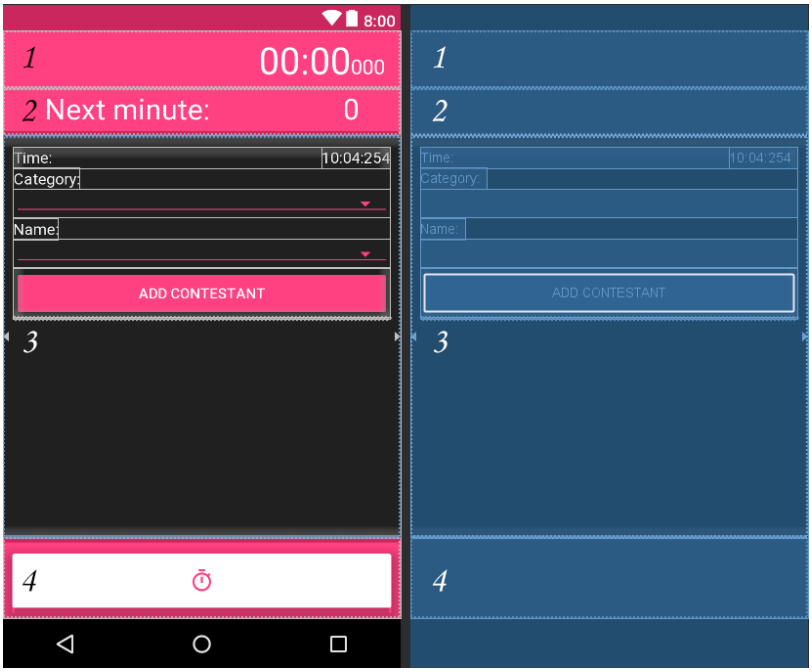
3.5 Layout

Jak již bylo řečeno dříve, Layout[12] je vlastně ViewGroup obsahující další view (obrázek 3.2). Máme ovšem několik typů layoutů. Tím nejzákladnějším je LinearLayout[13].

LinearLayout skládá své potomky podle svého nastavení v jednom směru buď pod sebe (do sloupce – vertical), nebo vedle sebe (do řádku – horizontal). V případě, že by velikost layoutu přesahovala velikost obrazovky, tak se automaticky vytvoří posuvníky a je tak možné vidět i zbytek obsahu za hranou obrazovky. Důležitou součástí LinarLayoutu je možnost nastavit každému potomku vlastní váhu (layout_weight). Na základě těchto vah se rozděluje místo v layoutu. Základní hodnota váhy je 0. Pokud má prvek váhu 0, nebo váhu nemá, zabere tolik místa, kolik potřebuje pro svoje správné vykreslení. Při vyšší váze se pak šířka nebo výška (dle orientace layoutu) poměrově rozdělí. Mějme například dva prvky v layoutu, které bychom řadili vedle sebe (orientation: horizontal). První prvek s vahou 1 a druhý prvek s vahou 2, pak první prvek zabere jednu třetinu šířky layoutu a druhý prvek dvě třetiny šířky layoutu. Rozložení 2 ku 3 je možné vidět na obrázku 5.4 vpravo.

LinearLayout však umí rozložit prvky pouze do řádků nebo sloupců. Pokud bychom chtěli vytvářet složitější uspořádání, budeme muset určovat pozici prvků jiným způsobem. V této době je doporučeno využít ConstraintLayout[14] (obrázek 3.3). Pro určení pozice každého prvku musíme nastavit horizontální a vertikální omezení (constraint). Proto musíme vždy znát, případně nastavit, id prvku, ke kterému se odkazujeme. Tímto způsobem můžeme zarovnávat prvky relativně vedle sebe, ale také přes sebe (typicky FAB[15] – FloatingActionButton).

[12] https://developer.android.com/guide/topics/ui/declaring-layout.html
[13] https://developer.android.com/guide/topics/ui/layout/linear.html
[14] https://developer.android.com/training/constraint-layout/index.html
[15] https://developer.android.com/reference/android/support/design/widget/FloatingActionButton.html



Obrázek 3.3: ContstraintLayout zobrazen v prostředí Android Studio (zobrazení „Design + Blueprint“). ConstraintLayout se rozkládá přes celou obrazovku a obsahuje čtyři view. **1.** Toolbar, **2.** horizontální LinearLayout, **3.** a **4.** vertikální LinearLayout. Každý layout zabírá celou šířku obrazovky. Layouty č. **1**, **2** a **4** mají pevně danou výšku. Výška layoutu č. **3** je dána zbývajícím místem mezi layouty č. **2** a **4**, protože nemá nastavenou pevnou velikost. Omezení jednotlivých view jsou: **1.** bez omezení – toolbar je vždy navrchu, **2.** horní hrana zarovnána se spodní hranou toolbaru (**1**), **3.** levá a pravá hrana zarovnána k levé a pravé hraně rodiče – (tedy k hraně obrazovky), horní hrana zarovnána ke spodní hraně view č. **2**, spodní hrana zarovnána na horní hranu view č. **4**, **4.** spodní hrana zarovnána se spodní hranou rodiče (obrazovky).

3.6 TextView, EditText a Button

Při vytváření uživatelského rozhraní se můžeme setkat s velkým množstvím komponent. Všechny komponenty mají jako předchůdce (ať už přímého nebo vzdáleného) třídu View. Jednou takovou komponentou je i TextView[16]. Jedná se o základní komponentu, která slouží hlavně k zobrazení textu uživateli. Je možné nastavit různé parametry jako třeba velikost, barvu nebo styl písma. Z TextView dále dědí další komponenty, které jsou už složitější

Často využívaným prvkem pro vstup textových informací je EditText[17] (obrázek 3.4). Při focusu (výběru komponenty) se na základě nastaveného typu vstupu (inputType) zobrazí softwarová klávesnice (pokud nemá zařízení klávesnici hardwarovou). Rozložení klávesnice je dané typem vstupu, přičemž není možné zadat jiné hodnoty, než tento typ definuje. Pokud je například vstup nastaven jako číslo (number), zobrazí se uživateli klávesnice číselníku a je možné zadávat pouze čísla. Tyto typy je dále možné mezi sebou kombinovat anebo vytvářet vlastní seznam povolených znaků. Můžeme nastavit předem vyplněný text

[16] https://developer.android.com/reference/android/widget/TextView.html
[17] https://developer.android.com/reference/android/widget/EditText.html

nebo nápovědu – text, který je viditelný pokud není pole vyplněno. Oběma textům můžeme nastavit zvlášť barvu, velikost nebo styl.

Další velmi často využívanou komponentou je Button[18] (tlačítko). Tlačítka se vyžívají pro spuštění určité události. Tu je potřeba naprogramovat a za pomoci listeneru tlačítku nastavit. Nejčastěji využívaným listenerem je „OnClickListener“ reagující při stisku tlačítka, či „OnLongClickListener“, reagující na dlouhý stisk nebo podržení. Tlačítku je možné nastavit barvu pozadí, barvu, velikost a styl textu, případně přidat obrázek (drawable). Jiným typem tlačítka je ImageButton[19]. Liší se hlavně absencí textu na tlačítku. Anebo také FAB – FloatingActionButton, standartně kulaté tlačítko, které se „vznáší“ nad ostatním obsahem.

3.7 Adapter

Pro zobrazení dat seznamu, kde není omezen počet prvků, musíme použít strukturu zvanou Adapter[20]. Adaptér hraje roli prostředníka mezi rodičovským layoutem a jeho potomky (data item). Rodičovský layout se nestará o samotné vytváření nebo mazání svých potomků. Tuto činnost zastřešuje právě adaptér, který je zodpovědný za zobrazení správného počtu prvků, nastavení jejich hodnot a chování, a také za udržení jejich aktuálnosti. Při změně zobrazovaných dat je tedy nutné upozornit adaptér (metodou „notifyDataSetChanged()“), který změny zohlední.

Existuje několik jednoduchých adaptérů, které dědí z třídy BaseAdapter[21]. Tyto adaptéry ale dokáží zobrazit jen jednoduchá data. Například ArrayAdapter[22], který vytváří jednotlivá view pomocí volání „toString()“ nad každým prvkem vstupní kolekce. Z toho je patrné, že není možné použít předem připravený layout, který zobrazuje více informací nebo uživateli zprostředkovává větší kontrolu nad každou položkou.

Z toho důvodu se běžně implementují vlastní adaptéry, jejichž chování musíme sami naprogramovat. Jde ale hlavně o nastavení parametrů view uvnitř layoutu, který reprezentuje data item. Díky vlastním adaptérům můžeme zobrazovat informace z kolekcí složitých datových struktur, přidávat do layoutu tlačítka, upravovat viditelnost view uvnitř layoutu podle zobrazovaných dat nebo nastavovat listenery pro komponenty v layoutu.

3.8 RecyclerView

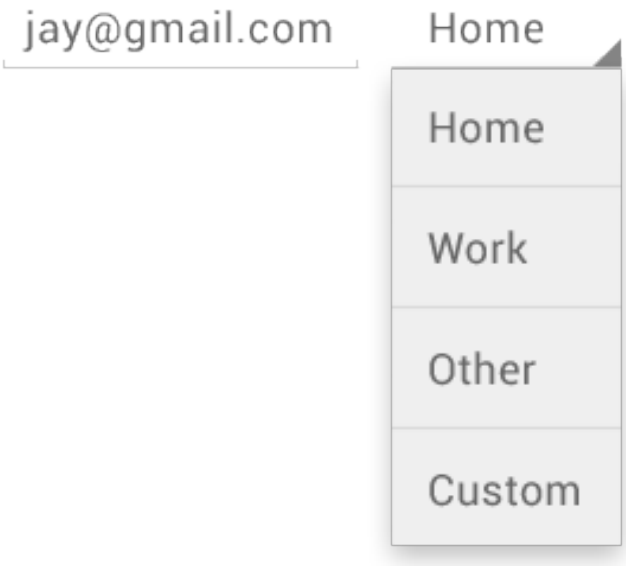
Doporučovanou metodou pro zobrazování rozsáhlých kolekcí dat je využití RecyclerView[23]. RecyclerView využívá k namapování dat do view třídu RecyclerView.Adapter[24]. Tento adaptér pracuje velmi podobně jako adaptéry popsané výše. Stále je zodpovědný za správné nastavení vykreslovaného view. Co ale RecyclerView dělá jinak, je napojení dapteru na LayoutManager[25].

[18] https://developer.android.com/reference/android/widget/Button.html
[19] https://developer.android.com/reference/android/widget/ImageButton.html
[20] https://developer.android.com/reference/android/widget/Adapter.html
[21] https://developer.android.com/reference/android/widget/BaseAdapter.html
[22] https://developer.android.com/reference/android/widget/ArrayAdapter.html
[23] https://developer.android.com/reference/android/support/v7/widget/RecyclerView.html
[24] https://developer.android.com/reference/android/support/v7/widget/RecyclerView.Adapter.html
[25] https://developer.android.com/reference/android/support/v7/widget/RecyclerView.LayoutManager.html

LayoutManager ve spolupráci s adaptérem zjišťuje, které item view je ještě viditelné, a které ne. Adaptér na základě těchto informací vytváří pouze view, která jsou viditelná na obrazovce (ať už úplně nebo jen částečně), dále jedno view před a jedno view za nimi (budou vidět pokud se v RecyclerView posuneme). Díky této recyklaci view můžeme zobrazovat velké množství dat s malým množstvím aktivních view. To má pozitivní dopad hlavně na výkon a využitou paměť telefonu.

3.9 Spinner

Spinner[26] (obázek 3.4) slouží k výběru jedné hodnoty z pevně zadaného seznamu. Vždy zobrazuje právě vybranou položku. Při klepnutí na spinner se otevře dropdown nabídka se všemi hodnotami, z nichž je možné vybírat. Seznam jednotlivých hodnot můžeme nastavit v layoutu ve kterém spinner vytváříme, nebo za pomoci adaptéru. Při použití adaptéru nám postačí i ArrayAdapter. Výhodou je možnost upravovat seznam položek ve spinneru za běhu aplikace.



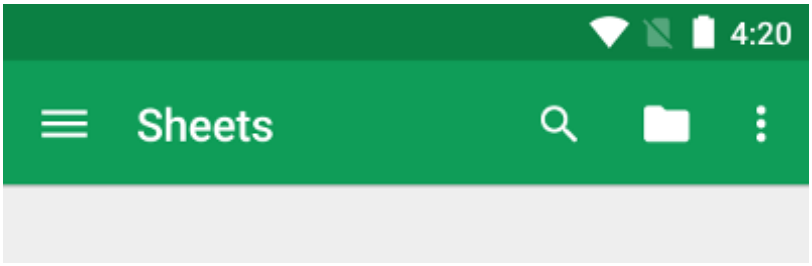
Obrázek 3.4: Ukázka komponent EditText (vlevo) a Spinner (vpravo).

Spinneru, jako každému jinému view, je možné nastavit řadu listenerů. Často používaným je „onItemSelected“, který vykoná svoji akci při výběru hodnoty ze seznamu. K vybrané položce lze přistupovat několika způsoby. Můžeme využít pozici vybraného prvku v seznamu nebo jeho hodnotu. Dále je také možné pracovat s celým vybraným view.

[26] https://developer.android.com/guide/topics/ui/controls/spinner.html

3.10 Toolbar

Toolbar[27] je ovládací a informační prvek aplikace. Zpravidla bývá umístěn na horní hraně okna aplikace, ale je možné zanořit ho do struktury view. Toolbar vychází z komponenty ActionBar[28] (obrázek 3.5), kterou dále rozšiřuje. ActionBar zobrazuje ikonu a název aplikace a options menu (více v následující kapitole) – pokud ho aktivita obsahuje.



Obrázek 3.5: ActionBar obsahující (zleva): tlačítko menu, titulek, Options menu. Options menu se dvěma viditelnými položkami – ikona lupy, složky a skryté položky – ikona tří svislých teček vpravo.

Toolbar ovšem přidává další funkce. Můžeme zobrazovat tlačítko navigace, logo aplikace, které přesahuje rozměry toolbaru, titulek a podtitulek, vlastní view a speciální action menu[29] – zobrazující ikony dle jejich nastavení.

V aktivitě je nutné toolbar nastavit pomocí „setactionBar()“ metody. Tento toolbar bude poté osazen už zmíněným options menu a tlačítkem navigace.

3.11 Options menu

V Options Menu[30] zobrazujeme seznam možných akcí, které se vztahují k aktuální aktivitě. V Android 2.3.x (API 10 a menší) se zobrazilo při klepnutí na tlačítko menu ve spodní části obrazovky. Od verze Android 3.0 (API 11 a vyšší – dnes již většina Android zařízení) se zobrazuje po klepnutí na ikonku v AppBaru[31] (Toolbar). Budeme se věnovat právě této variantě.

V toolbaru (obrázek 3.5) můžeme zobrazovat pouze ikonku menu (tři svislé tečky) nebo i ikonky (texty) položek menu, které pak v zobrazeném menu nejsou. Pokud jsou všechny položky menu zobrazeny na toolbaru, nebude se už vykreslovat tlačítko menu. Položkám menu (komponenta „item“) můžeme nastavit, kdy se budou na toolbaru zobrazovat. To uděláme nastavením patřičné hodnoty vlastnosti „showAsAction“. Výchozí hodnotou je „never“. Ta určuje, že položka nebude nikdy na samotném toolbaru, ale zobrazí se v seznamu po kliknutí na tlačítko menu. Opakem je možnost „allways“, kdy se ikonka položky zobrazí vždy a to bez ohledu na možné překrývání s dalšími prvky toolbaru. Pokud chceme zobrazovat ikonku na toolbaru a není nezbytně nutné, aby byla vždy viditelná, je lepší variantou využít možnost „ifRoom“. Ikonka se pak zobrazí pouze v případě, že je na toolbaru místo, jinak zůstává skryta a zobrazí se v seznamu stejně jako při nastavení „never“. V toolbaru je

[27] https://developer.android.com/reference/android/widget/Toolbar.html
[28] https://developer.android.com/reference/android/app/ActionBar.html
[29] https://developer.android.com/reference/android/widget/ActionMenuView.html
[30] https://developer.android.com/guide/topics/ui/menus.html#options-menu
[31] https://developer.android.com/training/appbar/index.html

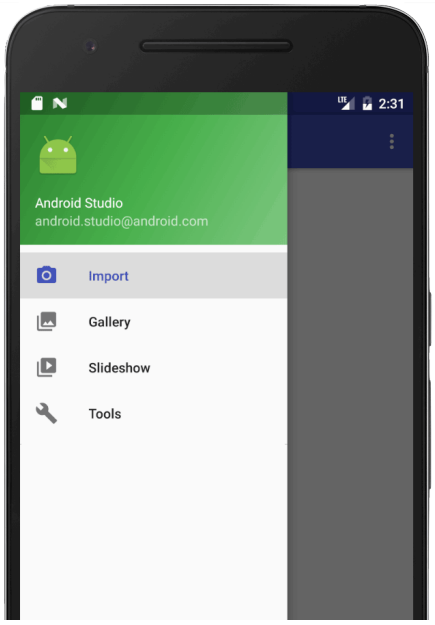
ovšem možné zobrazit i text – titulek (vlastnost „title“). Ke zmíněným hodnotám přidáme pomocí „pipe“ (znak „|“) hodnotu „withText“.

Samotný seznam položek je možné vytvořit jak za pomoci kódu přímo v aktivitě, tak i preferovanou variantou pomocí XML souboru. Tento seznam se přepsáním metody „onCreateOptionsMenu“ nastaví aktivitě. Samotné akce jednotlivých položek menu je nutné definovat v metodě „onOptionsItemSelected“. Vybranou položku určíme pomocí vráceného a námi definovaného ID položky.

Může nastat situace, kdy potřebujeme změnit položky options menu. K tomu slouží metoda „invalidateOptionsMenu“. Ta má za cíl znovu připravit a následně vytvořit options menu.

3.12 Navigation Drawer

Navigation Drawer[32] slouží jako hlavní navigační menu aplikace. Standardně je ho možné zobrazit pomocí klepnutí na tři vodorovné linky v levé části toolbaru nebo pomocí tažení od levého okraje obrazovky směrem doprostřed. Tuto funkcionalitu poskytuje DrawerLayout[33]. DrawerLayout obsahuje přesně dvě view. Důležité je i jejich pořadí. Prvním je layout, který bude obsahovat vše, co je viditelné, pokud je Navigation Drawer schovaný. Druhým je view samotné navigace. Je doporučeno použít NavigationView[34], které je k tomuto účelu předurčeno.



Obrázek 3.6: Navigation Drawer – zeleně podbarvená hlavička následovaná seznamem položek navigace

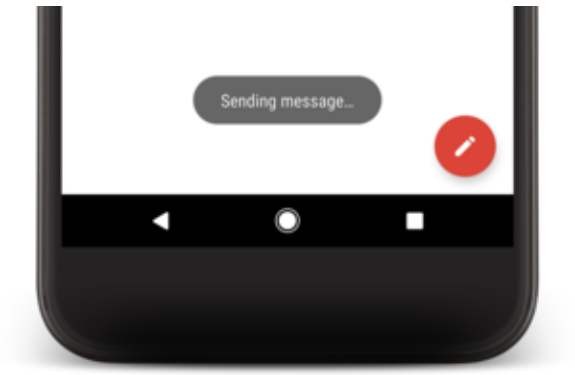
Samotná práce s navigací je pak podobná jako s Options Menu. Je třeba definovat položky navigace pomocí XML souboru. Dále je důležité nastavit aktivitě činnosti, které se provedou při klepnutí na položku. K tomu slouží metoda „onNavigationItemSelected“.

[32] https://developer.android.com/training/implementing-navigation/nav-drawer.html
[33] https://developer.android.com/reference/android/support/v4/widget/DrawerLayout.html
[34] https://developer.android.com/reference/android/support/design/widget/NavigationView.html

Odlišností je možnost přidat vlastní layout pro hlavičku vykreslovanou na vrcholu navigace. Dále je třeba přidat Navigation Drawer v kódu aktivity a nastavit tlačítko v toolbaru.

3.13 Toast

Toast[35] slouží pro jednoduché upozornění uživatele. Toast zobrazí text na šedém pozadí, standardně v dolní části obrazovky. Doba, po kterou je toast vidět, je dána jeho nastavením (obvykle se používá konstanta „Toast.LENGTH_SHORT“ nebo „Toast.LENGTH_LONG“).



Obrázek 3.7: Toast oznamující odesílání zprávy

[35] https://developer.android.com/guide/topics/ui/notifiers/toasts

Kapitola 4

Uživatelé

4.1 Charakteristika uživatele

S rostoucím výkonem mobilních zařízení se stále více činností přesouvá z počítačů na mobilní telefony a tablety. Tím také roste počet uživatelů preferujících tyto platformy. Zároveň se mobilní telefony a tablety stávají hlavním informačním nástrojem mnoha uživatelů[7]. Díky tomu se k aplikaci může snadno dostat jak dvacetiletý trenér žákovského týmu vytrvalostních běžců, tak i důchodkyně, která pro ně ve svém volném čase organizuje závody. Oba používají telefony s Androidem a oba používají stejnou aplikaci – naše stopky.

Z toho důvodu musíme cílit na široké spektrum uživatelů všech věkových skupin a pohlaví. Aplikace musí nabízet jednoduché uživatelské rozhraní, ale zároveň velké množství funkcí. Tyto funkce musí být ale snadno pochopitelné. Musí být jasné, co se stane po stisknutí tlačítka. Uživatel by měl být schopný vykonávat potřebné akce intuitivně a s minimálním úsilím. Nechceme, aby si uživatelé museli pokládat otázky typu: „Jak se udělá to a to?“ a „Jak mám opravit tohle?“.

Ať už budeme měřit závod na běžecké dráze za tělocvičnou, nebo na kraji lesa daleko od civilizace, chceme poskytovat stejné a spolehlivé služby. Chceme se spolehnout, že správně změříme časy všem závodníkům, že vyhodnotíme jejich pořadí a odměníme vítěze. Nechceme složitě přepisovat výsledky a časy z poznámkového bloku. Chtěli bychom je jednoduše, pomocí stisku tlačítka, sdílet s účastníky závodu.

4.2 Potřeby uživatele

Uživatel aplikace bude potřebovat vytvářet a ukládat určitá data o závodu. V následujícím seznamu jsou sepsány základní entity, se kterými bude uživatel pracovat. Obsah těchto entit je jen přibližný – zohledňuje požadavky uživatele.

- Závod
  - jméno
  - seznam závodníků a kategorií
  - čas odstartování
- Závodník
  - jednoznačná identifikace

  - jméno a poznámka
  - čas startu
- Kategorie
  - seznam závodníků
  - jméno
  - čas startu prvního závodníka
  - interval mezi závodníky
- Čas
  - čas odečtení
  - korekce
  - diskvalifikace

Závod reprezentuje skutečnou sportovní akci, závodník reálného člověka účastnícího se závodu a kategorie seskupuje závodníky do skupin. Čas pak reprezentuje dobu, která uběhla od odstartování závodu do průchodu závodníka cílem (odečtení času) a slouží k výpočtu celkového času závodníka.

Následuje seznam akcí a požadavků, které by uživatel mohl potřebovat. Konkrétními způsoby naplnění těchto požadavků se věnuje kapitola 5.

- Vytvoření, kopírování a odstranění závodu
- Přidání, odebrání a úprava závodníka
- Přidání, odebrání a úprava kategorie
- Rozdělení závodníků do kategorií
- Rozlosování startovních časů podle nastavení kategorie
- Spuštění časomíry s určitým časem
- Zvuková signalizace startu
- Přesné změření času
- Spárování času se závodníkem
- Úprava (korekce) a diskvalifikace času
- Zobrazení startovní listiny (startovky)
- Zobrazení aktuálních výsledků
- Export závodu, startovky a výsledků (Google Sheets)
- Import závodu (Google Sheets)

Kapitola 5

Návrh aplikace

V této kapitole se budu detailně věnovat návrhu uživatelského rozhraní aplikace a jednotlivých funkcí. Popíši zde, jakým způsobem jsem řešil problémy s návrhem spojené, jaké jsem vyzkoušel přístupy, co fungovalo a co se ukázalo jako slepá cesta.

Velkou výzvou při návrhu uživatelského rozhraní pro mobilní telefony je správné využití prostoru na obrazovce. V knize Mobile Usability[6] se Jakob Nielsen a Raluca Budiu věnují různým způsobům návrhu mobilních verzí webových stránek, ale i mobilních aplikací. U zařízení s menším rozměrem obrazovky poukazují na důležitost rozložení jednotlivých prvků. Je důležité využít prostor efektivně, ale zároveň nezahltit uživatele příliš velkým množstvím informací. Dále zdůrazňují nutnost zvolit vhodné ovládací prvky tak, aby bylo možné pohodlně, rychle a přesně ovládat aplikaci pouhým dotykem.

S podobným přístupem přichází i Steve Krug v knize Don't Make Me Think, Revisited[5]. Ten však vyzdvihuje jednoduchost uživatelského rozhraní jako celku, krátké a jasné popisky tlačítek a položek menu. Grafické rozdělení důležitých a méně důležitých informací. Každá položka by měla mít svůj význam a důvod existence.

S těmito přístupy souhlasím a v průběhu návrhu a testování se ukázala jako velmi platná. Na další aspekty uživatelské přívětivosti poukazuje ve svém článku Troy Parke[9]. Vyzdvihuje zde využívání jednotného vizuálního stylu pro nativní aplikace. Pro Android je to Material Design[2]. V našem případě se jedná hlavně o využití několika základních barev bez jakéhokoli vzoru a využití vlastnosti elevation – vlastnost layoutu, která opticky zvedne view směrem k ploše obrazovky a vrhá stín na view, která jsou níž.

5.1 O testování

Testování jednotlivých částí uživatelského rozhraní probíhalo na menší skupině lidí (obvykle 3–5). Dle reakce testujících jsem pak nevyhovující prvky upravil, nebo úplně přepracoval. Díky těmto rychlým testům jsem mohl návrh iterativně upravovat a zkoušet různé varianty.

Další částí testování bylo otestování celých obrazovek a orientace v zobrazovaných prvcích. Cílem bylo ujistit se, že uživatel ví, k čemu jaká část aplikace slouží a co zobrazuje. K tomuto účelu bylo nutné použít větší skupinu osob (přibližně 15). Mezi nimi pak byli i trenéři a rozhodčí. V této skupině bylo majoritní zastoupení uživatelů telefonů s os. Android, menší počet uživatelů os. iOS a Windows.

5.2 Propojení entit

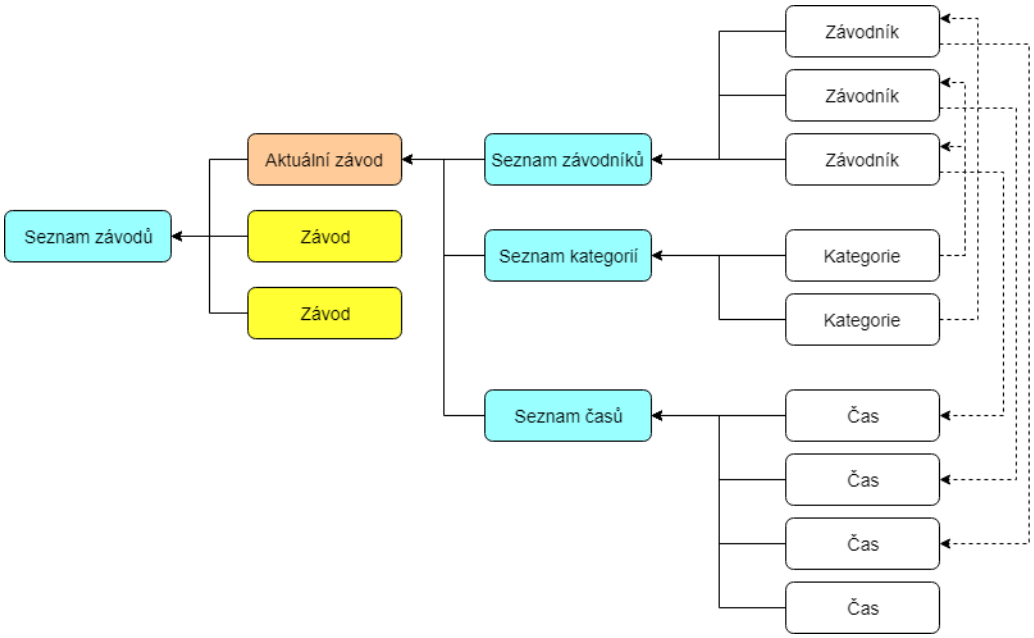
Před vlastním návrhem uživatelského rozhraní je třeba si ujasnit, jakým způsobem bude aplikace fungovat. Díky tomu zjistíme, jakým způsobem bude nutné pracovat s daty a kdy, kde a jak je bude nutné zobrazovat.

V kapitole 4.2 jsme již definovali základní entity, se kterými budeme pracovat. Je to čas, závodník, kategorie a závod. Na obrázku 5.1 je znázorněno základní schéma entit v aplikaci. Aplikace musí mít vždy vybraný jeden závod (aktuální závod). S tímto závodem může uživatel pracovat. Ostatní závody nejsou načteny a je možné pouze jejich přejmenování, duplikování, smazání a načtení.

Propojení entit podléhá následujícím pravidlům:

- Každý závodník má přesně jednu přidělenou kategorii
- Každý čas má přiděleného maximálně jednoho závodníka.

Z toho vyplývá, že není možné vytvořit závodníka, pokud neexistuje alespoň jedna kategorie. Dále je možnost přiřadit jednomu závodníkovi více časů. S touto variantou je počítáno pro budoucí rozšíření – měření kol (mezičasů). V současném návrhu tedy nepovolíme více časů přiřazených jednomu závodníkovi.



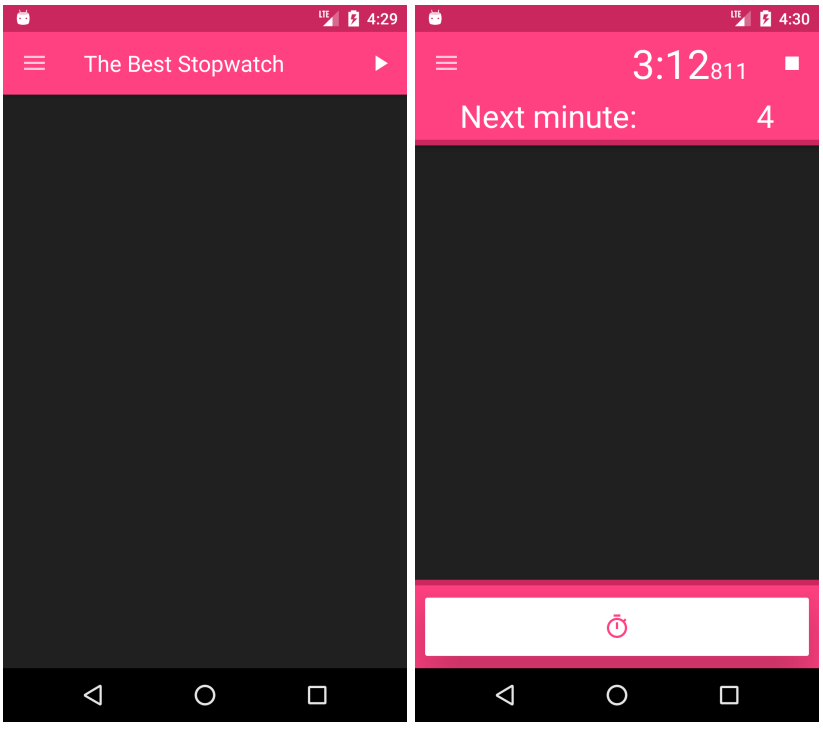
Obrázek 5.1: Diagram znázorňující propojení jednotlivých entit v aplikaci. Modře jsou značený seznamy, žlutě neaktivní závody, oranžově aktuální závod a bíle entity závodníků, kategorií a časů. Plná čára značí přímou vazbu (závod obsahuje tři seznamy, seznamy obsahují entity). Čárkovanou čarou je označen odkaz na entitu (závodník odkazuje na kategorii, čas odkazuje na závodníka).

5.3 Základní rozložení

Vzhled všech obrazovek (view) aplikace bude vycházet z aktuálního stavu závodu. Z pohledu rozložení obrazovky máme pouze dva stavy a to, že čas běží a čas neběží. Rozdíl mezi obrazovkami je znázorněn na obrázku 5.2. Nejdůležitějšími oblastmi jsou tlačítko pro vytvoření nového času a změna rozložení horní části s časem závodu.

Tlačítko na odečtení času je vždy na stejném místě, nikdy není překryto jiným objektem a jeho funkce se nemění. Díky tomu může uživatel zaznamenat čas v jakékoli aktivitě a při jakékoli další akci bez jejího přerušení. Pokud se pak uživatel nenachází na obrazovce se seznamem časů (hlavní obrazovka aplikace – obrázek 5.2), je upozorněn na vytvoření nového času toastem(kapitola 3.13). Na hlavní obrazovce není třeba oznamovat nový čas pomocí toastu, protože uživatel uvidí změnu přímo v seznamu časů.

Při návrhu tlačítka jsem vyzkoušel i variantu s použitím FAB. To se při testování ukázalo jako příliš malé a bylo snadné ho minout. Ani jeho výchozí umístění na pravé straně obrazovky nebylo při delším používání pohodlné. Posledním větším problémem bylo mírné splývání s položkami pod tlačítkem. Prodloužením a zvětšením tohoto tlačítka se problémy z části vyřešily, ale nezmizely. Z toho důvodu jsem zvolil variantu s tlačítkem po celé šířce obrazovky. To je možné stisknout v kterémkoli bodě a díky výraznému pozadí pod tlačítkem je snadno identifikovatelné.



Obrázek 5.2: Hlavní obrazovka aplikace. Na obrázku vlevo můžeme vidět základní obrazovku bez běžícího času. Obsahuje pouze toolbar s názvem aktuální aktivity. Napravo je obrazovka v druhém stavu – čas běží. V toolbaru je zobrazen aktuální čas závodu. Pod toolbarem je další řádek, na kterém je dle aktuální aktivity vypsán název aktivity, nebo číslo následující minuty. Na spodní hraně je velké tlačítko pro zaznamenání nového času. Tmavá oblast je vždy určena pro obsah aktuální aktivity.

Minuty a sekundy na toolbaru jsou vypsány velkým písmem. Milisekundy jsou vypsány menším písmem. Hodiny jsou záměrně vynechány. K tomuto kroku jsem přistoupil z důvodu snadnějšího výpočtu (z hlavy) výsledného času závodníka (startuje v celou minutu). Počet minut závodu tedy může přesáhnout číslo 60. Absence hodin také pomáhá zpřehlednit měnící se čas. Při testování se v tomto ohledu nevyskytly žádné potíže.

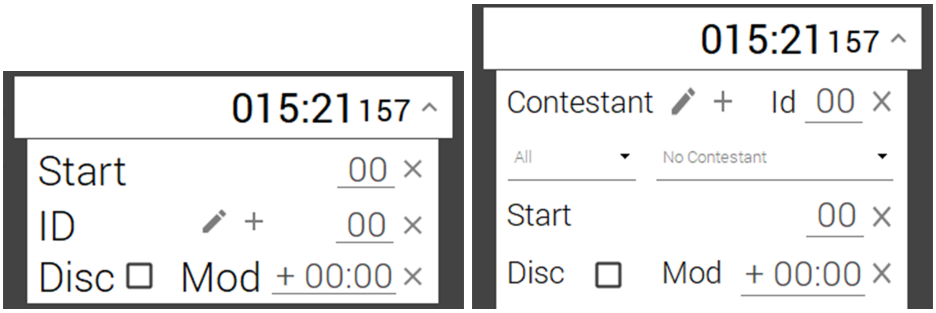
Ukazatel následující minuty byl přidán na základě vlastních zkušeností z pořádání závodu. Potvrdil jsem si, že při kontrole závodníka připraveného na startu je mnohem jednodušší podívat se na číslo na obrazovce, než číst aktuální čas a přičítat minutu. Toto se může zdát jako trivialita, ale organizátor se tak může soustředit na důležitější činnosti.

Testováním se ukázalo, že na hlavní obrazovce (se seznamem časů), která bude při závodě nejpoužívanější, je následující minuta vítaný prvek. Problém nastal v ostatních aktivitách, hlavně u seznamu závodníků a kategorií. Větší počet lidí měl problém s orientací. Nebyli si jistí, ve které aktivitě právě jsou (hlavně pokud jsem narušil jejich soustředění, například upozorněním na „přibíhajícího“ závodníka) a bylo pro ně jednodušší vrátit se na hlavní obrazovku a ujistit se, že byli tam, kde chtěli být. Proto jsem přistoupil k vypsání jména aktivity (stejně jako v případě, že neběží čas) namísto následující minuty. Při dalším testování se už takový problém neobjevoval.

5.4 Karta času

Karta času reprezentuje čas v seznamu časů. Slouží zejména ke spárování závodníka s časem a k úpravě času. Základní návrh spočívá v jednoduchém obdelníku s časem, který se při klepnutí „otevře“ a zobrazí ovládací rozhraní.

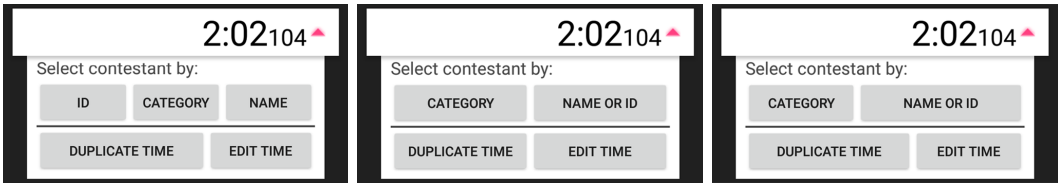
První navržená varianta (obrázek 5.3) sdružovala přiřazení závodníka a úpravu času do jednoho view. Z obrázků je vidět velké množství ovládacích prvků. Otestováním tohoto přístupu jsem zjistil, že pro seznámeného uživatele je přidání závodníka, úprava jeho startovního času nebo diskvalifikace velmi rychlá. Neseznámený uživatel měl však problém s velkým množstvím ovládacích prvků. Také se vyskytly problémy s přesností ovládání a nechtěnou změnou údajů. Z těchto důvodů vznikla druhá varianta (obrázky 5.4 a 5.5).



Obrázek 5.3: Původní návrh karty času. První verze (vlevo) přiřazovala závodníka podle jeho identifikátoru (ID), dovolovala upravit jeho startovní čas a také modifikovat a diskvalifikovat reprezentovaný čas. V druhé verzi přibyla možnost přiřadit závodníka výběrem pomocí spinnerů.

První varianta přinesla mnoho problémů. Z toho důvodu vznikla druhá varianta (obrázky 5.4 a 5.5). Upustil jsem od úpravy údajů přímo v kartě. Namísto toho jsem pro přidání závodníka vytvořil několik aktivit (podle způsobu přiřazení) a jednu aktivitu pro úpravu času. Dále jsem přidal tlačítko pro duplikaci času. Konečným výsledkem je karta se

čtyřmi tlačítky. Tím se razantně zmenšil počet ovládacích prvků. Také přestala existovat možnost náhodné změny údajů. S touto variantou neměli při testování problém ani méně zkušení uživatelé.

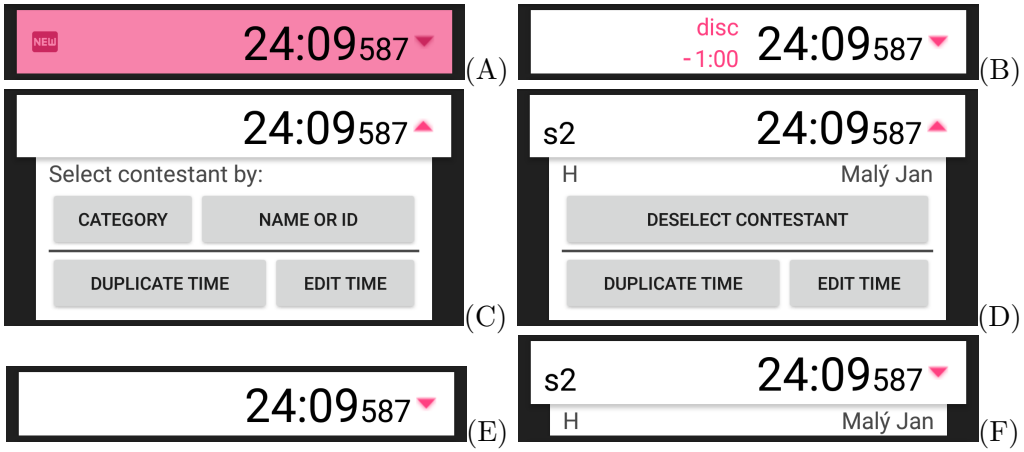


Obrázek 5.4: Posloupnost změn vzhledu druhé varianty karty času. Prvotní návrh (vlevo) počítal s variantou tří způsobů přiřazení závodníka k času (podle identifikátoru, kategorie nebo jména) – první řádek tlačítek. Další tlačítka slouží pro úpravu tohoto času a pro jeho duplikaci – druhý řádek. V průběhu implementace se ukázalo, že není třeba rozlišovat způsob přiřazení pomocí identifikátoru a jména. Z toho důvodu byly tyto operace sloučeny do jedné. Tato čtyři tlačítka (uprostřed) ovšem vytvářela mřížku a při více otevřených kartách nad sebou se obrazovka stávala nepřehlednou. To to se vyřešilo v poslední verzi (vpravo), kde se změnila velikost tlačítek, a tím se mřížka rozbila.

Základní princip s tlačítky se tedy osvědčil. Dalším krokem návrhu je definovat možné stavy, ve kterých se karta může nacházet. Tyto stavy jsou zobrazeny na obrázku 5.5. Šipka na pravé straně hlavičky (hlavní část karty s časem) znázorňuje možnost otevření a zavření karty. Nový čas je označen ikonou a po otestování byla hlavička pro lepší viditelnost podbarvena. Čas je vypsán stejným způsobem jako v toolbaru. Informace o diskvalifikaci nebo úpravě času je zobrazena malým barevným písmem vedle času. Jedná se pouze o informativní výpis a změna těchto hodnot se odehrává ve vlastní aktivitě. V hlavičce může být také zobrazen startovní čas přiděleného závodníka (například „s2“ – startovní čas „2“). K optickému rozdělení hlavičky a těla je využita vlastnost elevation a zúžení těla karty.

Tlačítka a texty v těle karty jsou méně výrazné, a tak nerozptylují uživatele. Celé tělo je viditelné pouze,tehdy pokud je karta otevřena. Částečně můžeme tělo vidět, pokud je karta zavřena a času je přidělen závodník. V tom případě je na těle karty zobrazen název kategorie a jméno závodníka.

Otevřít a zavřít kartu může pouze uživatel. Experimentoval jsem s automatickým zavřením karty po přiřazení závodníka, ale po návratu z aktivity nebylo jasné, který čas byl otevřen. To vedlo ke zmatení uživatele.



Obrázek 5.5: Poslední nově přidaný čas (A), upravený a diskvalifikovaný čas (B), otevřená karta bez přiřazeného závodníka (C), otevřená karta s přiřazeným závodníkem (D), zavřená karta bez přiřazeného závodníka (E), zavřená karta s přiřazeným závodníkem (F)

5.5 Výpočet času a vyhodnocení závodu

Při výpočtu výsledného času závodníka se můžeme inspirovat například pravidly orientačního běhu[3]. Čas se vypočte jako rozdíl cílového a startovního času závodníka. V našem případě je ještě třeba přičíst úpravu času. Ta může sloužit jak pro opravu času po špatném odečtení, tak i pro penalizaci nebo zvýhodnění závodníka.

Při zjišťování aktuálního času je vždy použita funkce systému, která vrací čas zařízení v milisekundách. Díky tomu není nutné implementovat vlastní systém hodin a předejdeme tak možnosti zrychlení nebo zpomalení v důsledku orchestrace vykonávaných vláken.

Pro celý výpočet musíme použít několik hodnot. Jsou to:

A Čas startu závodu – entita závod – systémový čas v čase 0

B Čas startu závodníka – entita závodník – počet minut od času 0

C Odečtený čas – entita čas – systémový čas v okamžiku odečtení času organizátorem

D Úprava času – entita čas – korekce času (kladná nebo záporná) v minutách a sekundách, kterou nastaví uživatel

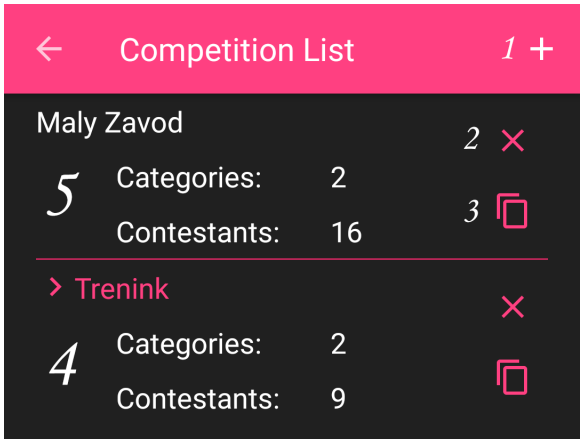
Pomocí těchto hodnot jsme schopní přesně určit čas, který závodník strávil na trati (**X**) od času svého startu, a to podle vzorce:

$$C - A - B + D = X$$

Pořadí závodníků je pak určeno pro každou kategorii zvlášť. Všem závodníkům se vypočítají časy. Poté se určí pořadí dle výsledných časů. Nakonec je třeba přesunout diskvalifikované závodníky (závodníci s diskvalifikovaným časem) na konec. Důvod diskvalifikace určuje pořadatel závodu.

5.6 Závody

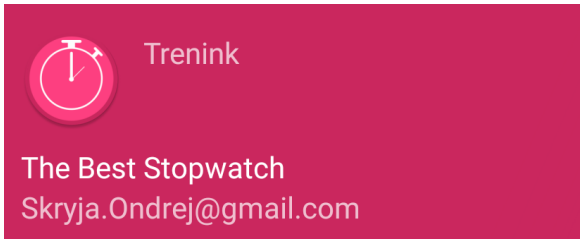
Všechny závody uložené v paměti telefonu jsou zobrazeny v seznamu závodů (obrázek 5.6). Odtud je možné vytvořit, načíst, zkopírovat nebo smazat závod. Položka závodu obsahuje jméno závodu, počet kategorií a počet závodníků.



Obrázek 5.6: Část obrazovky seznamu závodů. **1.** nový závod, **2.** smazat závod, **3.** duplikovat závod, **4.** načtený (aktivní) závod, **5.** uložený (neaktivní) závod (jde načíst).

Při vytvoření nového závodu je možné zadat pouze název závodu. Po potvrzení se závod vytvoří a automaticky načte. Název načteného závodu je v seznamu vyznačeno barevně. Také se zobrazí v hlavičce navigačního menu (obrázek 5.7). Při prvním spuštění aplikace se automaticky vytvoří nový závod. Každý vytvořený závod neobsahuje žádné kategorie, závodníky ani časy.

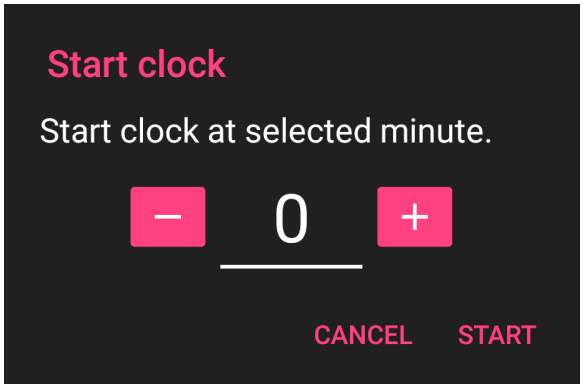
Načtení a úprava je možná po kliknutí na položku vybraného závodu (zobrazí se dialog s výběrem akce). Načtení závodu vrátí uživatele zpět na hlavní obrazovku. Při duplikaci závodu je nutné vybrat, jakým způsobem se má závod duplikovat. Je možné vytvořit úplnou kopii – čas startu, kategorie, závodníci a časy, nebo částečnou kopii – kategorie a závodníci. Úplná kopie dovoluje vyzkoušet destruktivní změny (smazání závodníka nebo kategorie), zatímco částečná kopie se hodí k rychlému znovuvytvoření závodu s původními závodníky (například při tréninku – není nutné znovu zadávat závodníky a kategorie).



Obrázek 5.7: Hlavička navigačního menu – zobrazuje aktuálně načtený závod (Trenink)

Odstartování závodu je možné pouze v případě, že závod neprobíhá. Tlačítko pro odstartování se nachází na toolbaru hlavní obrazovky a je vždy viditelné (obrázek 5.2 vlevo). Klepnutím na toto tlačítko se objeví jeden ze dvou dialogů. Pokud jde o závod, který nebyl nikdy spuštěn, zobrazí se dialog s výběrem minuty, ve které bude čas začínat (obrázek 5.8). Výchozí hodnota je 0. Můžeme zadat kladný i záporný čas. To to je výhodné, pokud chceme

například startovat v konkrétní dobu, nebo potřebujeme vykonávat nějakou předstartovní proceduru.



Obrázek 5.8: Dialog výběru minuty startu

Druhý startovní dialog se zobrazí, pokud byl závod odstartován a poté zastaven. Dialog nabízí několik možných akcí:

- pokračovat v závodu – hodiny se spustí a čas bude pokračovat jako by nebyl zastaven (stav závodu zůstává nezměněn)
- restartovat závod – nenávratně se smaže seznam časů a zobrazí se dialog s nastavením minuty startu
- duplikovat – vytvoří se částečná kopie závodu (původní závod může být znovu načten) a zobrazí se dialog s nastavením minuty startu.

Po odstartování se mění základní rozložení obrazovek a tlačítko pro odstartování se transformuje na tlačítko zastavení (ikona trojúhelníku na ikonu čtverečku). Při zastavení hodin se změní rozložení obrazovek a tlačítko stop se změní na tlačítko start.

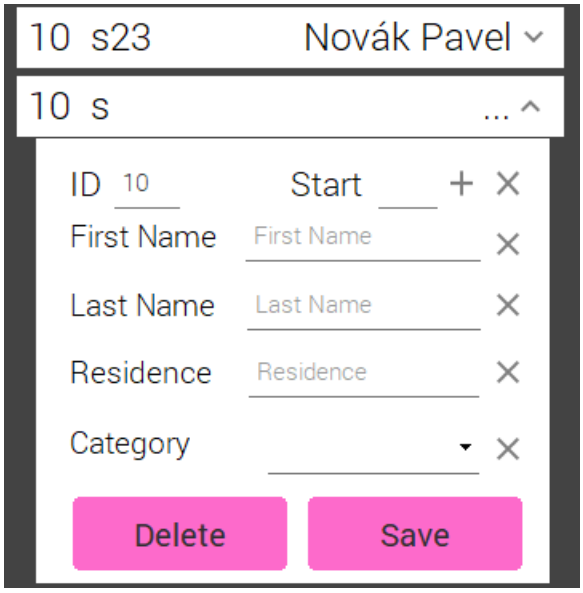
5.7 Závodníci

U návrhu seznamu závodníků byla původní myšlenka podobná jako u seznamu časů. Návrh se skládal z karty závodníka (obrázek 5.9). Stejně jako u seznamu časů, tak i zde se měla veškerá interakce se závodníkem odehrávat na této kartě. Při testování ovšem nastal stejný problém jako dříve. Karta byla složitá a na malém prostoru bylo příliš mnoho informací. Na základě těchto zkušeností jsem upustil od seznamů tvořených podobnými kartami (s výjimkou karty času na hlavní obrazovce).

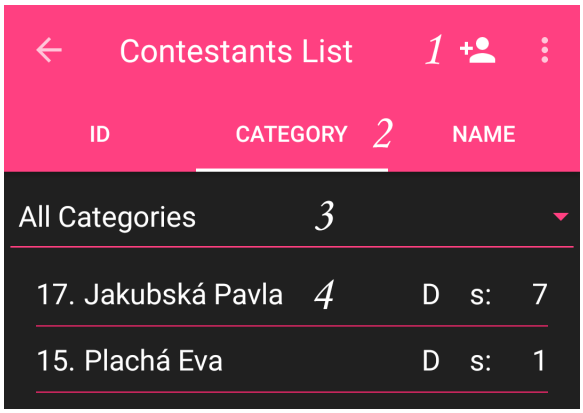
Konečným výsledkem je seznam s jednořádkovými položkami (obrázek 5.10). Každá položka zobrazuje základní informace o závodníkovi – identifikátor, jméno, kategorii a startovní čas. V seznamu závodníků je použit TabLayout[1]. Díky němu je možné posunutím doleva nebo doprava přecházet mezi jednotlivými záložkami. Každá záložka zobrazuje stejný seznam závodníků (změny se promítnou do všech záložek), ale s jiným způsobem řazení. Je možné seřadit závodníky podle identifikátoru, kategorie a jména. V případě řazení podle kategorie je možné vybrat, která kategorie se má zobrazit. Klepnutím na položku se otevře nová aktivita úpravy závodníka. Dlouhým podržením na kterékoli položce se u všech

[1] https://developer.android.com/reference/android/support/design/widget/TabLayout

závodníků zobrazí zaškrtávací pole a provede se výběr položky. Takto můžeme vybrat libovolný počet závodníků. Ti zůstávají vybráni i ve chvíli, kdy změníme záložku. V options menu můžeme zahájit nebo ukončit výběr závodníků, smazat vybrané závodníky a vylosovat startovní časy závodníkům. Aktivitu pro přidání nového závodníka je možné zobrazit klepnutím na ikonu v options menu. Vyzkoušel jsem i variantu s použitím FAB. Zde jsem ale narazil na problém s překrýváním části informací v delších seznamech, a proto jsem ji nepoužil.



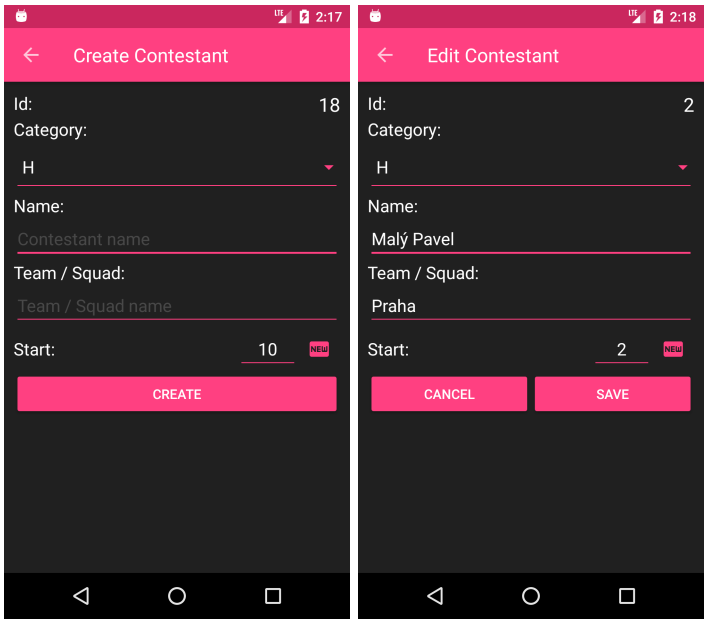
Obrázek 5.9: Prototyp karty závodníka. Nahoře je zavřená karta vytvořeného závodníka. Dole je otevřená karta nového závodníka.



Obrázek 5.10: Seznam závodníků. **1.** tlačítko pro přidání nového závodníka, **2.** záložky (TabLayout) určující způsob řazení v seznamu (vybrána záložka kategorie), **3.** spinner pro výběr zobrazené kategorie – pouze v záložce kategorie, **4.** položka seznamu reprezentující závodníka.

Vytvoření a úprava závodníka sdílí stejnou aktivitu (obrázek 5.11). Vzhled této aktivity je upraven v závislosti na tom, jakým způsobem byla spuštěna (zda byl předán identifikátor závodníka). V případě úpravy závodníka se vyplní všechna pole hodnotami patřícímu

upravovanému závodníkovi. Identifikátor (ID) získá závodník při svém vytvoření a není možné ho změnit. Jedná se o jediný unikátní identifikátor, který závodník má. Výběr kategorie je realizován jako spinner obsahující všechny kategorie. Oproti původním polím v kartě závodníka se jméno zadává jako jeden řetězec znaků. Tím ubyl jeden parametr. Pole „Team / Squad“ obsahuje informaci o skupině, ke které závodník patří. Toto pole není povinné a jeho hodnota má pouze informativní význam. Zároveň je viditelná pouze v exportech do Google Sheets (kapitola 5.11). Díky tomuto parametru si pak účastníci závodu mohou například lépe spojit jméno svého soupeře se jménem města nebo klubu, který reprezentuje. Poslední vyplňovanou hodnotou je čas startu. Čas startu určuje přesnou minutu, kdy bude závodník startovat. Tuto hodnotu může uživatel zadat sám, nebo může po stisku tlačítka „NEW“ získat volný čas. Ten je dán nastavením vybrané kategorie. Čas se zobrazí v dialogovém okně a uživatel si může vybrat, zda ho chce přidělit. Tento čas se vždy hledá od času 0 nebo od aktuálního času závodu (pokud závod běží). Aplikace zároveň vybírá čas tak, aby vyplnila možné mezery mezi závodníky.

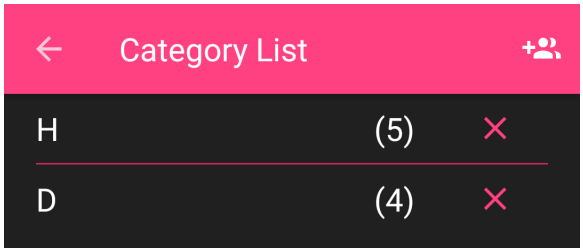


Obrázek 5.11: Obrazovka pro vytvoření (vlevo) a úpravu (vpravo) závodníka

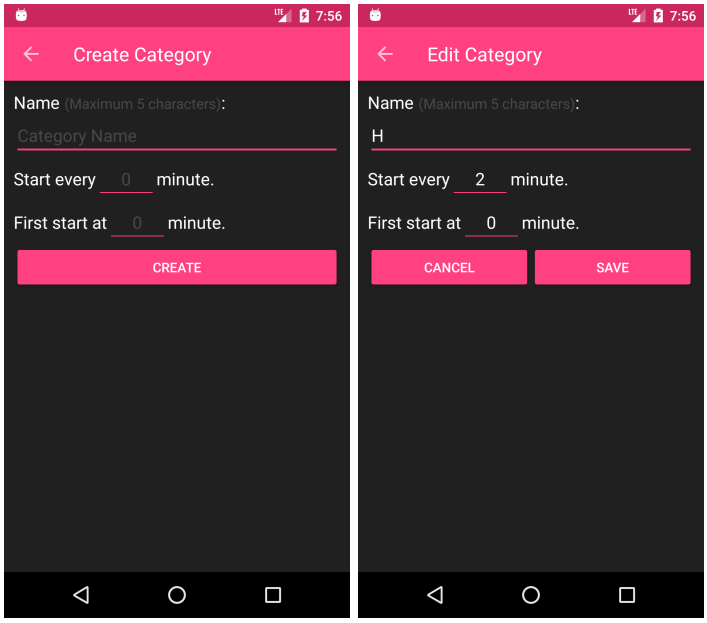
5.8 Kategorie

Podobně jako seznam závodníků, tak i seznam kategorií je tvořen jednořádkovými položkami. Také tlačítko pro vytvoření nové kategorie je řešeno stejným způsobem (obrázek 5.12).

Úprava a vytvoření nové kategorie opět sdílejí jednu aktivitu (obrázek 5.13). Název kategorie musí být ve vybraném závodě unikátní a jeho délka je omezena na 5 znaků. Toto omezení je hlavně kvůli omezenému místu při vypsání v seznamu závodníků, ve spinnerech a v exportovaných a importovaných tabulkách. Další vyplňované hodnoty slouží k rozlosování závodníků do kategorií. Je to čas startu prvního závodníka v kategorii a interval mezi starty závodníků. Obě hodnoty jsou zadávány v celých minutách. Díky tomuto nastavení je možné startovat jak intervalově, tak hromadně. Změny tohoto nastavení ovšem neovlivní startovní časy závodníků. To se stane až při přepočítání časů vyžádaným uživatelem.



Obrázek 5.12: Seznam kategorií. Položka seznamu obsahuje (zleva) jméno, počet závodníků a tlačítko pro smazání kategorie.



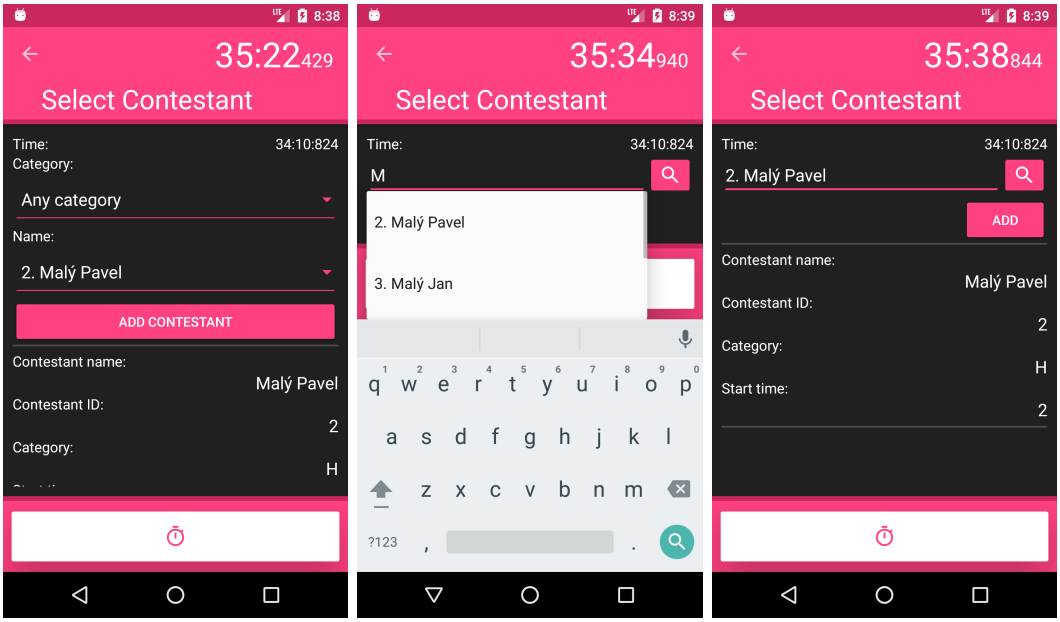
Obrázek 5.13: Obrazovka pro vytvoření (vlevo) a úpravu (vpravo) kategorie

Zajímavá situace nastává při smazání kategorie. Jedním z pravidel aplikace je, že závodník musí mít přidělenou kategorii. Prostým odstraněním kategorie by se toto pravidlo porušilo. Tento problém jsem vyřešil přidáním dialogu při smazání kategorie, který uživateli nabídne dvě možnosti. Smazat kategorii a všechny přiřazené závodníky, nebo všechny závodníky automaticky přesunout do jiné vybrané kategorie.

5.9 Přiřazení závodníka k času

V kapitole 5.4 jsou popsány dva způsoby, jak přidat závodníka k času. Aktivity, které tuto akci zprostředkovávají, jsou na obrázku 5.14. V obou variantách se při úspěšném vyhledání zobrazí informace o závodníkovi. Tak si uživatel může zkontrolovat, zda přiděluje čas správnému závodníkovi.

V případě výběru pomocí kategorie je nalezení závodníka jednoduché. Je využita kombinace dvou spinnerů. První slouží pro výběr kategorie (jedné nebo všech). Druhý pro výběr závodníka z kategorie vybrané prvním spinnerem. Při změně vybrané kategorie se aktualizuje spinner závodníků. Při aktualizaci spinneru závodníků se aktualizuje detail vybraného závodníka.



Obrázek 5.14: Přidání závodníka pomocí kategorie (vlevo), jména nebo identifikátoru (uprostřed – vyhledávání, vpravo – vyhledáno).

Výběr pomocí textu je o něco složitější. Uživateli se při zadávání textu zobrazují možní závodníci (pomocí AutoCompleteTextView[2]). Klepnutím na navrhovaného závodníka se provede doplnění textu a vyhledání. Při úspěšném vyhledání se zobrazí informace o závodníkovi a tlačítko pro přidání. Pokud je ovšem zadán řetězec, který neodpovídá připraveným možnostem (jménům závodníkům nebo jejich identifikátoru) a nejde tedy vyhledat, zobrazí se informace o špatně zadaném řetězci.

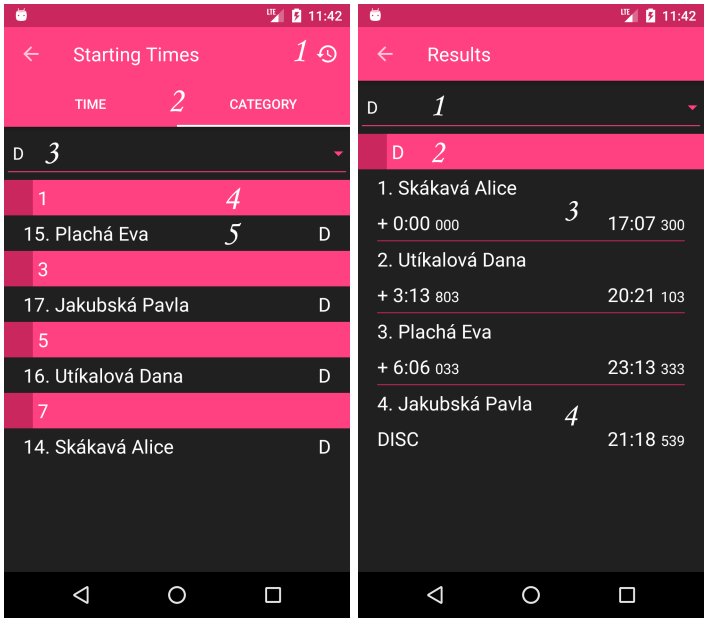
Závodník s přiřazeným časem nesmí být přiřazen k jinému času. V obou způsobech přiřazení jsou nabízeni pouze závodníci bez přiděleného času.

[2] https://developer.android.com/reference/android/widget/AutoCompleteTextView

5.10 Startovní a výsledková listina

Seznam startovních časů je možné zobrazit v jedné ze dvou záložek (obrázek 5.15). První záložka zobrazuje časy všech závodníků, druhá zobrazuje časy vybrané kategorie. V obou případech jsou v seznamu zobrazeny dva různé druhy položek. Jedna položka s časem startu, po níž následují položky závodníků s tímto startovním časem. V této aktivitě je možnost rozlosovat startovní časy.

Výsledky (obrázek 5.15) jsou zobrazeny buď po jednotlivých kategoriích, nebo po prvních třech nebo pěti závodnících v každé kategorii (toto zobrazení je vhodné pro vyhlašování výsledků). Podobně jako u startovky je seznam složen ze dvou typů položek seznamu. Jedna zobrazuje název kategorie a je následována položkami s pořadím, jménem a časem závodníka a jeho ztrátou na vedoucího závodníka. Diskvalifikovaní závodníci se řadí na konec kategorie dle svých časů. Způsob výpočtu výsledných časů závodníků je popsán v kapitole 5.5.



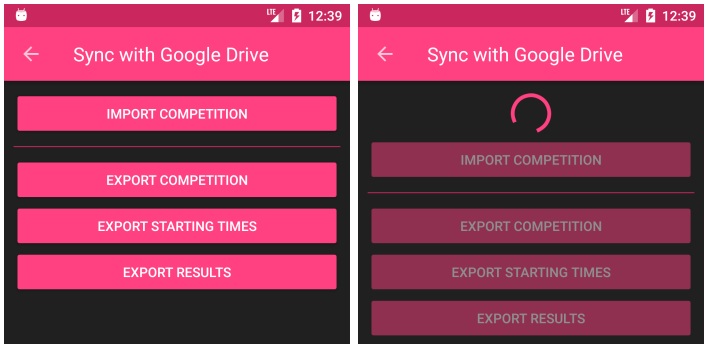
Obrázek 5.15: Startovka (vlevo) obsahující: **1.** tlačítko pro rozlosování startovních časů **2.** záložky pro zobrazení časů startujícího a časů podle kategorie **3.** spinner pro výběr kategorie **4.** položka seznamu označující startovní čas **5.** položka seznamu označující závodníka. Výsledky (vpravo) obsahující: **1.** spinner pro výběr kategorie **2.** položku seznamu označující kategorii **3.** závodníka s výsledným časem a ztrátou na nejlepšího **4.** diskvalifikovaného závodníka.

5.11 Google Drive a Google Sheets

Pro spojení s Google Drive a Google Sheets je třeba nejprve zkontrolovat, zda je zařízení připojeno k internetu, zda uživatel udělil požadovaná povolení a zda je uživatel přihlášen ke Google účtu. Tyto kontroly je nutné provést pokaždé, když se ke službám Google přistupuje. V průběhu kontrol jsou tlačítka pro synchronizaci (obrázek 5.16) skrytá a je vidět pouze rotující ProgressBar[3]. Po úspěšném ověření všech požadavků se tlačítka zobrazí. Pokud

[3] https://developer.android.com/reference/android/widget/ProgressBar

kontrola nebyla úspěšná, zobrazí se text s touto informací a tlačítko pro opětovnou kontrolu. Pokud uživatel není přihlášen ke Google účtu, je mu zobrazeno přihlašovací okno (toto okno poskytuje Google API).



Obrázek 5.16: Obrazovka pro synchronizaci s Google Drive. Vlevo jsou odblokovaná tlačítka. Vpravo jsou zablokovaná tlačítka při synchronizaci.

Po kliknutí na tlačítko požadované akce jsou všechna tlačítka zablokována a po celou dobu operace je na obrazovce viditelný progressBar. Po ukončení operace je uživateli zobrazen toast s informací o výsledku operace a tlačítka jsou opět aktivní.

Při exportu závodu, startovky nebo výsledků je v kořenovém adresáři Google Drive uživatele vytvořen nový Google Sheet soubor s požadovanou tabulkou (obrázek 5.17). Pro import je nutné najít v Google Drive (okno pro vyhledávání poskytne Google Drive API) soubor s exportovaným závodem. Pokud je soubor v pořádku (odpovídá struktura souboru a nejsou žádné kolize v datech), je vytvořen a následně načten tento závod.

The Best Stopwatch - Trenink								
Blue cells will be automatically filed. Do not insert or edit any values.								
ID	Category (max 5 characters):		H		Time			Final Time
	First Start:	0	Interval	2				
	Name		team / Squad		Start	Finish	Mod.	DISC	
2	Malý Pavel		Praha		4	24:25:411	0:00:000	false	20:25:411
3	Malý Jan				6	24:10:630	0:00:000	false	18:10:630

(A)

The Best StopWatch				
Brno - 24. 2. 2018 - Krátká Trať				
Category H				
Place	Name	Team / Squad	Time	Loss
1.	Smutný Jan	Příbram	16:23:154	
2.	Malý Jan		18:10:630	+ 1:47:476

(B)

The Best StopWatch			
Brno - 24. 2. 2018 - Krátká Trať			
Category H			
Order	Name	Team / Squad	Start
1.	Smutný Jan	Příbram	0
2.	Malý Jan		2

(C)

Obrázek 5.17: Ukázky hlaviček tabulek pro export dat. **A** – hlavička pro export a import závodu, **B** – hlavička pro výsledky závodu, **C** – hlavička pro startovní listiny

Kapitola 6

Implementace

6.1 Kotlin vs. Java

Při výběru programovacího jazyka, ve kterém jsem aplikaci psal, jsem neváhal. Výběr padl na Kotlin, a to hned z několika důvodů. Kotlin je plně kompatibilní s Javou. Je možné použít oba programovací jazyky dohromady. To také znamená, že Kotlin může využívat většinu Java knihoven. V Kotlinu je možné psát kratší a přehlednější kód než v Javě (ukázka kódu 6.1 a 6.1). Kotlin ovšem nabízí ještě mnohem víc[10][4].

```
// Java
  button.setOnClickListener(new View.OnClickListener(){
    @Override
    public void onClick(View v){
      doSomething();
    }
  });

// Kotlin
  button.setOnClickListener { doSomething() }
```

Ukázka kódu 6.1: Zápis stejného listeneru v Javě a Kotlinu

```
// Java
  TextView text = (TextView) findViewById(R.id.myTextView);
  text.setText("Hello World");

// Kotlin
  myTextView.setText("Hello World")
```

Ukázka kódu 6.2: Nastavení textu do TextView v Javě a Kotlinu

6.2 Ukládání dat

Data o právě načteném závodě jsou uložena v operační paměti. Díky tomu je přístup k nim velmi rychlý. Při ukončení aplikace se ale takto uložená data přepíší. Je tedy třeba uložit data trvaleji. K tomu je využita SQLite[1] databáze. Tato databáze je uložena v paměti zařízení a přístup k ní má pouze aplikace, která ji vytvoří. Zde jsou pak uložena veškerá data všech závodů. Při akci, která mění nebo vytváří data, se změna nejprve zanese do operační paměti a poté do databáze. Důvodem je časová náročnost zápisů do databáze. Pro práci s databází jsem použil třídu SQLiteOpenHelper[2], která slouží k vytváření a upravování databáze a také pomáhá implementovat CRUD[3] operace.

6.3 Autentizace služeb Google

Pro připojení ke Google Drive a Google Sheets je třeba nejprve zaregistrovat aplikaci v Google Cloud Platform[4]. Poté je potřeba k aplikaci přidat Credentials (slouží pro připojení a povolení námi vybraných API). V credentials se zadává námi vygenerovaný SHA-1 certifikát. Tímto certifikátem se podepisuje přeložený APK soubor. Je proto nutné vytvořit credentials pro „debug“ i „release“ sestavení. Po vytvoření credentials je třeba vybrat, které API budeme používat. V našem případě je to Google Drive API a Google Sheets API.

V tuto chvíli by tedy měla mít aplikace přístup k vybraným službám Google. Je tedy potřeba provést kontroly popsané v kapitole 5.11. Pro volání Google API pod účtem přihlášeného uživatele je třeba po jeho přihlášení získat „GoogleAccountCredential“ sloužící pro ověření identity pří volání Sheets API[5]. Dále pak „DriveResourceClient“, který slouží pro volání Drive API[6].

[1] https://developer.android.com/reference/android/database/sqlite/package-summary
[2] https://developer.android.com/reference/android/database/sqlite/SQLiteOpenHelper
[3] Create, Read, Update, Delete
[4] https://console.cloud.google.com/apis/
[5] https://developers.google.com/sheets/api/
[6] https://developers.google.com/drive/

Kapitola 7

Závěr

Cílem mé bakalářské práce bylo vytvořit Android aplikaci pro organizaci závodu a měření časů. Výsledkem je zdarma stažitelná aplikace v Obchodě Play pod jménem „The Best Stopwatch“[1].

Aplikace umí vytvářet závodníky a rozdělit je do kategorií, rozlosovat startovní časy a změřit časy jednotlivým účastníkům závodu. Dále stanoví pořadí závodníků a výsledky přehledně zobrazí. Celý závod je možné importovat a exportovat s použitím služby Google Sheets. Stejným způsobem je možné sdílet startovní listinu a výsledky závodu.

V průběhu vývoje jsem vyzkoušel několik návrhů uživatelského rozhraní. Nakonec jsem implementoval to nejlépe vyhovující, které bylo vybráno na základě opakovaného testování na skupině uživatelů. Věřím že se mi tak podařilo vytvořit přehlednou a jednoduše ovladatelnou aplikaci, která navíc poskytuje uživateli mnoho užitečných funkcí a usnadní mu práci při pořádání závodů.

Nakonec jsem vytvořil prezentační video a plakátek, které stručně představují vytvořenou aplikaci, její hlavní funkce a přínosy pro uživatele.

Aplikaci The Best Stopwatch bych v budoucnu rád rozšířil o možnost měřit časy kola, na co je již aplikace připravena. Dále bych ji rád lokalizoval do více jazyků a přidal možnost výběru z více barevných palet. Dalším možným rozšířením by mohlo být vytvoření dalších typů závodů, například týmový a štafetový závod.

[1] https://play.google.com/store/apps/details?id=com.skryja.ondrej.bptimer

Literatura

[1] *Android dokumentace*. [Online; navštíveno 24.04.2018].
URL https://developer.android.com/docs/

[2] *Mategial Design Guidelines*. [Online; navštíveno 24.04.2018].
URL https://material.io/

[3] *Pravidla orientačního běhu*. Platná od 1. 2. 2018, [Online; navštíveno 24.04.2018].
URL
http://www.orientacnibeh.cz/upload/dokumenty/sekce-ob/pravidlaob-18.pdf

[4] Hearn, M.: *Why Kotlin is my next programming language*. *https://medium.com*, 6 2015.

[5] Krug, S.: *Don't make me think, revisited : a common sense approach to web usability*. San Francisco : New Riders, 2014, ISBN 978-0321965516.

[6] Nielsen, J.; Budiu, R.: *Mobile usability*. Berkeley, CA : New Riders, 2013, ISBN 978-0-321-88448-0.

[7] Salz, P.: *Good Mobile Design Is Good Business*. *EContent*, ročník 37, č. 2, 12 2014: str. 8, ISSN: 1525-2531.

[8] Shafirov, M.: *Kotlin on Android. Now official*.
https://blog.jetbrains.com/kotlin/, 5 2017.

[9] Troy, P.: *How to Design Native Mobile Apps*. *https://www.uxhow.com/*, 11 2015.

[10] Vinther, M.: *Why you should totally switch to Kotlin*. *https://medium.com*, 5 2017.

[11] Yener, M.; Dundar, O.; Hellman, E.: *Expert Android Studio*. United States: Wrox Press Ltd, 2016, ISBN 1119089255.